Identifying Languages using Support Vector Machines

Gábor Loerincs
Depto. de Computación
Universidad Simón Bolívar
Caracas, Venezuela
gloerinc@usb.ve

Sandra Zabala
Depto. de Computación
Universidad Simón Bolívar
Caracas, Venezuela
szabala@usb.ve

Willmer Pereira
Escuela de Informática
Universidad Católica Andrés Bello
Caracas, Venezuela
wpereira@usb.ve

Abstract

The current spread of Internet makes it possible to find online information in a great number of languages. Tasks involving automatic processing of this information may require the identification of the language in which it is written.

In this work we explored the use of Support Vector Machines, a relatively new classification technique from the machine learning field, for generating language classifiers to distinguish English from any other language in written texts.

We have trained and tested Support Vector Machines, representing documents as feature vectors based on the occurrence of character sequences in texts (N-Grams). Our empirical results show that the Support Vector Machines technique is suitable for the language identification task, building very accurate linear classifiers.

Keywords: Written Language Identification, Support Vector Machines, Inductive Learning

1 Introduction

Language Identification (language ID for short) is a specific instance of the more general problem of classification, where some language category has to be assigned to a natural language sample. Although traditionally spoken language ID has received more attention than written language ID, the situation has changed in the last few years. The main causes of such change are:

- the current spread of Internet makes it possible to find online information in a great number of languages and, as a consequence,
- a previous identification of the language may be required by tasks such as indexing FAQs, searching the Web and filtering news, which involve automatic processing of information.

Previous work has been done for written English, Spanish and even multilingual identification, using different techniques for the classification task. Some approaches use information about short words such as those presented by Kulikowski [11] and Ingle [9]. Others use methods based on

N-Grams, either of words, such as Batchelder [1], or characters, such as Cavnar and Trenkle [4] and Cowie et al. [6]. There are also techniques that use the independent probability of letters and the joint probability of letter combinations for deriving a Markov model, such as the one presented by Dunning in [8].

We focused our work on the written language ID problem, exploring the use of Support Vector Machines, a relatively new classification technique from the machine learning field, for generating language classifiers able to distinguish English from any other language.

The entire English ID process can be divided in three phases: first, language characterization; second, classifier training and third, classifier testing.

Figure 1 presents the general process of English characterization. It means generating a profile from a featuring data set composed of only English documents.

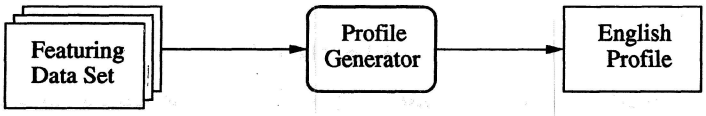

Figure 1: English profile generation.

Figure 2 outlines the training phase. It consists of building vectors representing documents in a training data set and using these vectors to generate a binary English classifier. The training set is composed of English and non-English documents.

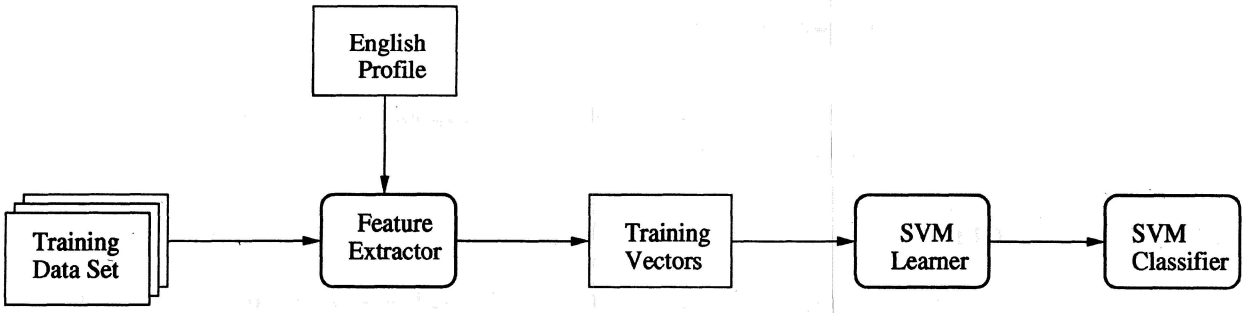

Figure 2: Training phase for the English ID process.

Figure 3 shows the testing phase, where new documents in a testing data set are used to evaluate the accuracy of the generated classifier.

In this work, the three data sets (featuring, training and testing) were taken from the soc.culture newsgroup hierarchy of the Usenet.

The rest of this paper is organized as follows: a general description of the Support Vector Machines technique is presented in section 2; in section 3 we describe in detail the English profile generation process; section 4 comprises the design and results of the experiments carried out during the training and testing phases; and finally, the conclusions and directions for future work are presented in section 5.

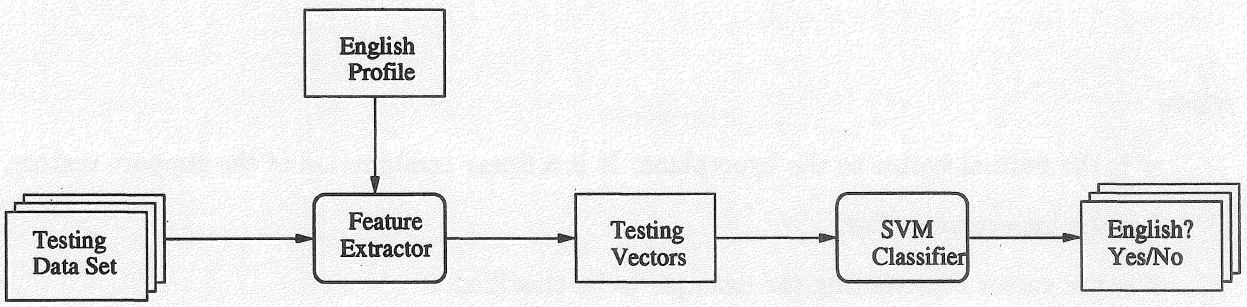

Figure 3: Testing phase for the English ID process.

2 Support Vector Machines

Support Vector Machines (SVMs for short) constitute a learning technique mainly used for the classification task. Although they were proposed by Vapnik in 1979, it is since few years ago when they have gained popularity and that is why we consider them a "relatively new" technique [2, 5, 15, 3]. SVMs are well founded in terms of the computational learning theory and open to theoretical analysis and understanding.

2.1 Definition

In its simplest linear form a SVM is a hyperplane that separates a set of positive examples (representing category $+1$) from a set of negative ones (representing category -1) with maximum margin (see Figure 4). The examples closest to the hyperplane are called Support Vectors.

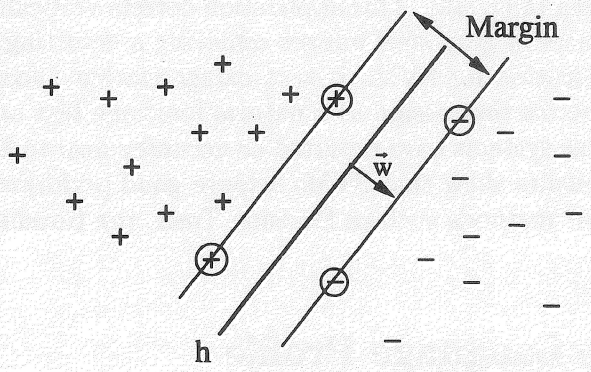

Figure 4: A linear Support Vector Machine h. Support Vectors are marked with circles.

The objective of training a linear SVM is to learn a classifier from the examples, i.e. to find a representation for the hyperplane, which only uses support vectors in its formulation.

The general formula of a linear SVM is the following:

$$u = \vec{w} \cdot \vec{x} - b \tag{1}$$

where

$\vec{w}$ is the normal vector to the hyperplane. It is a linear combination of the support vectors.

b is the hyperplane offset.

$\vec{x}$ is the vector representing the example to be classified.

$sgn(u)$ determines the category in which $\vec{x}$ will be classified.

For linearly separable problems, maximizing the margin is equivalent to solve the following optimization problem:

$$\begin{array}{ll} \text{Minimize:} & \frac{1}{2}||\vec{w}||^2 \\ \text{subject to:} & y_i(\vec{w} \cdot \vec{x_i} - b) \geq 1 \ \forall i \end{array}$$

where x_i is the ith training example and y_i is the category to which x_i belongs (-1 or $+1$) [1].

Finding $\vec{w}$ and b requires the solution of a Quadratic Programming problem. Many methods for solving QP problems are very slow for large problems. We implemented the algorithm proposed by Platt [13], which breaks the original QP problem into a series of the smallest possible QP problems and analytically solves them.

2.2 Applications

There are several examples of real-world applications of SVMs. One of them is the face detection system developed by Osuna et al. [12]. This application detects vertically oriented and unoccluded frontal views of human faces in gray-level images achieving a detecting rate as high as 97.1%.

Another kind of application for SVMs is text categorization. Joachims [10] and Dumais et al. [7] have developed systems that assign to a natural language text one or more topic categories based on its content. These systems have reported an accuracy near to 87% working with up to 118 topic categories. These results show that SVMs achieve good performance on text categorization tasks, outperforming other methods such as Decision Trees, the Rocchio algorithm and k-Nearest Neighbors [10].

3 Creating the Language Profile

As it was mentioned in section 1, it is necessary to create the English profile, when dealing with the language ID task. Two steps can be distinguished in the profile generation process: (1) choosing the document representation units suitable for the learning algorithm and (2) establishing a criterion to select the set of features that will conform the profile.

[1] For not linearly separable problems Cortes and Vapnik [5] and Boser et al. [2], have proposed formulations and extensions to learn non-linear classifiers.

3.1 N-Grams as Document Representation Units

Document representation has a direct impact on the generalization accuracy of the learning system. Typical document representation units, such as word stems [14], are based on the concept of *word*. These units can have the inconvenient of making the feature selection process language dependent. We chose N-grams of characters as document representation units to overcome this problem.

An N-Gram is a sequence of N characters. We build N-Grams from a string by extracting all the possible sequences of N contiguous characters from it, discarding digits and punctuation (except apostrophes). In our work we used N-Grams of different lengths simultaneously.

Table 1 presents some N-Grams that can be extracted from the string `that old theater`.

N	N-Grams
1	`_,t,h,a,o,l,d,e,r`
2	`_t,th,ha,at,t_,_o,ol,ld,d_,he,ea,te,er,r_`
3	`_th,hat,at_,_ol,old,ld_,d_t,the,hea,eat,ate,ter,er_`
4	`_tha,that,hat_,at_o,t_ol,_old,old_,ld_t,d_th,_the,` `thea,heat,eate,ater,ter_`

Table 1: Some N-Grams of the string `that old theater`. The character "_" represents a blank.

3.2 Feature Selection

By choosing N-Grams as the representation units we are dealing with an attribute-value representation of documents, where attributes are N-Grams. It is necessary to determine which values have to be associated to these attributes. As we are trying to characterize a language it is not enough to express that a given N-Gram is present or not in a document (binary approach). Associating the frequency of each N-Gram in the document gives more information to the classifier.

It is also necessary to determine which attributes have to be considered, because taking all the possible N-Grams may lead to a very high-dimensional feature space. To do this selection, the following steps were performed (Cavnar and Trenkle [4]):

- Parse a featuring data set, consisting of only English documents, to eliminate extra blanks, digits and punctuation (except apostrophes).
- Scan the parsed result, generating all possible N-Grams for several values of N.
- Associate to each N-Gram found its frequency in the featuring data set.
- Sort the N-Grams by its frequency value in descendent order.
- Delete the N-Gram "_" (considered not too representative).

The first k N-Grams in the sorted list are the attributes used for document representation. This set of attributes will conform the language *profile*. For example, consider the featuring data

t	h	a	e	_t	th	at	_th

Figure 5: Profile of dimension $k = 8$ built using the phrase `that old theater` as the featuring data set and the N-Grams presented in Table 1.

set constituted by the phrase `that old theater`. Figure 5 shows a profile of dimension $k = 8$, generated using the N-Grams presented in Table 1.

In order to set the values of k and N, several experiments were performed over a featuring data set of about 750.000 characters. Figure 6 shows the distribution of N-Grams frequency according to their ranking after the sorting process. This distribution shows that N-Grams ranked above 300 are less likely to appear in an English document than N-grams ranked below 300.

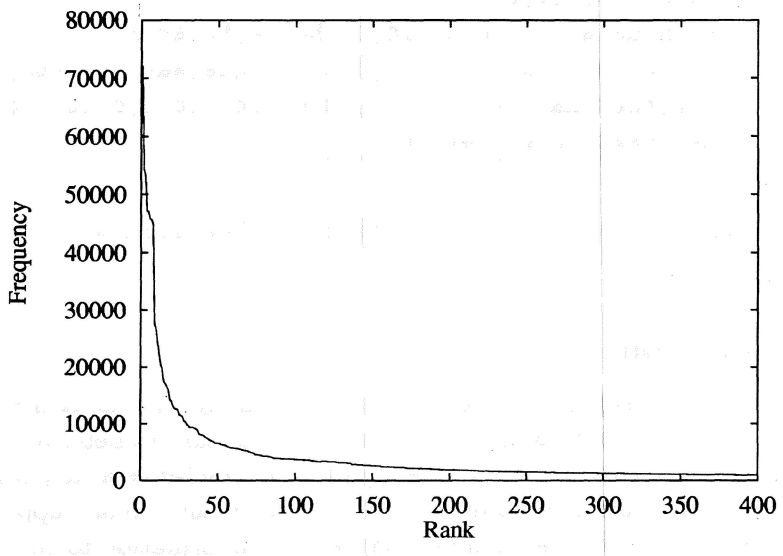

Figure 6: N-Gram frequencies by rank in the featuring data set.

We initially set $k = 300$ but in the classification experiments we tried with smaller values to study the impact of reducing the profile dimension in the accuracy of the classifiers. The results are presented in the next section.

Regarding the possible values of N, it is necessary to say that we only considered profiles where N ranges from a minimum to a maximum value (referred as $<min\text{-}max>$-profiles). In order to find the most convenient minimum and maximum values for N, we created profiles with different ranges. We decided to include 1-Grams in each profile to incorporate the letter distribution in English as a possible feature. Table 2 presents the distribution of N-Grams in four different profiles of dimension 300 with N ranging from 1 to 3,4,5 and 6 respectively. Entry (i,<1-j>) represents the percentage of N-grams of size i in the <1-j>-profile.

These percentages show that N-Grams of size grater than 4 do not contribute significantly to the language profile. As N-Grams of size 5 and 6 represent less than the 3% of the profiles, we decided to work with profiles that include N-Grams of up to 4 characters.

	Profiles			
N	<1-3>	<1-4>	<1-5>	<1-6>
1	8.70%	7.60%	7.60%	7.60%
2	55.30%	52.00%	51.70%	51.70%
3	36.00%	31.70%	30.70%	30.30%
4	-	8.70%	7.60%	7.60%
5	-	-	2.40%	2.40%
6	-	-	-	0.40%

Table 2: Percentage of N-Grams in different <*min-max*>-profiles of dimension 300 taking as featuring data set a collection of English samples from the soc.culture newsgroup hierarchy of the Usenet.

4 Classification

Two activities were performed in order to explore the suitability of the SVM technique for the language ID task: first, training classifiers and second, testing them. As Figures 2 and 3 show, both activities have in common the feature extraction process.

4.1 Feature Extraction

So far we have established the features (N-Grams) that describe the English language, i.e. its profile. Every document, in the training and testing data sets, can be represented as a vector whose components correspond to these features. Each component has associated, as value, the frequency of its corresponding N-Gram in the document. The process of associating values to features is known as feature extraction. Figure 7 shows the vector that represents the document `my mother's brother is my uncle`, using the profile presented in Figure 5.

Feature	`t`	`h`	`a`	`e`	`_t`	`th`	`at`	`_th`
Value	2	2	0	3	0	2	0	0

Figure 7: Vector that represents the document `my mother's brother is my uncle`, using the profile presented in Figure 5.

4.2 Training Classifiers

The training phase was accomplished by using a training data set of about one million characters. The amount of English samples was approximately the same of non-English samples. The non-English samples were written in 26 different languages.

We extracted features from every document in the training data set, using a <1-4>-profile of dimension 300, and then proceeded to train the SVM. We started training with the linear version. As we got a training accuracy of 100%, we trained three more SVMs using <1-4>-profiles of

dimension 200, 100 and 50, obtaining the same accuracy. Such results suggested that the English ID problem is linearly separable and made us decide not to try with a more complex version of SVMs.

It is worth to say that, although the training data set had more than 1200 samples, the number of support vectors was between 25 and 33 in all the experiments. It means that less than the 3% of the training samples are needed to build the SVM classifier. As a consequence, the calculation of the general formula (1), presented in section 2, is not expensive. The SVM classifier execution time is $\Theta(d)$, where d is the profile dimension.

4.3 Testing

During the testing phase we carried out several experiments with the four classifiers trained. We designed five testing data sets, of about 400 documents each, using English and non-English samples. Table 3 presents a description of each data set.

Data Set	Description
Test600	Documents of 600 characters (about 80 words).
Test300	Documents of 300 characters (about 40 words).
Test150	Documents of 150 characters (about 20 words).
Test75	Documents of 75 characters (about 10 words).
TestAny	Documents of assorted sizes (unrestricted number of words).

Table 3: Testing data sets.

Table 4 shows the experimental results.

	Classification accuracy (%)				
Prof. Dim.	Test600	Test300	Test150	Test75	TestAny
300	100.00	99.00	98.43	97.34	99.52
200	100.00	99.00	98.16	96.37	99.76
100	100.00	99.00	97.90	95.40	99.27
50	100.00	98.33	97.90	94.43	99.27

Table 4: Experimental results.

We can observe that for a given data set, accuracy decreases as the profile dimension is diminished [2]. However, this accuracy loss does not seem too significant. In the worst case, we got a difference of about 3% (classifying Test75 data set with profiles of dimension 300 and 50). Working with smaller profiles could be useful in real-time applications where the classification of medium-sized documents has to be done as fast as possible.

[2]We noticed the abnormal increase in the classification accuracy using profile dimension 200 with the TestAny data set. Examining the misclassified documents, we found one written half in English half in Flemish. We think that this fact and the elimination of N-Grams from the profile could explain the anomaly.

We can also observe that the classifiers behave quite well when dealing with small documents, yielding an average accuracy of 95.9% with documents of about 10 words.

Finally, the accuracy obtained with the TestAny data set (over 99%) suggests that the SVM technique is suitable for Internet applications, where the document size is not pre-established.

5 Conclusions and Future Work

This work explored the use of the Support Vector Machines technique for the written English identification task. As a result, we learned linear SVM classifiers that are very accurate, even with short documents and/or low-dimension English profiles. This suggests that the technique is suitable for the proposed task.

Besides, the feature selection and feature extraction processes used, were based on N-Grams, which are language independent and less expensive than word-based techniques, where the system has to perform word processing and needs to have detailed knowledge about the particular language each document is written in.

We believe that the framework introduced in this paper can be easily extended to build multilingual ID classifiers and it is our next assignment. It will allow us to compare the SVM classifiers accuracy with other existing multilingual ID classifiers. If similar results to those obtained here are achieved with multilingual classifiers, it is possible to think of using them in real applications, such as the Internet related ones.

References

[1] E. Batchelder. A learning experience: Training an artificial neural network to discriminate languages. Technical Report, 1992.

[2] B. Boser, M. Guyon, and V. Vapnik. A training algorithm for optimal margin classifiers. In *Proceedings of the Conference on Computational Learning Theory (COLT)*, pages 144–152, 1992.

[3] C. Burges. A tutorial on Support Vector Machines for pattern recognition. *Data Mining and Knowledge Discovery*, 2(2):1–47, 1998.

[4] W. Cavnar and J. Trenkle. N-Gram-Based Text Categorization. In *Proceedings of the Third Annual Symposium on Document Analysis and Information Retrieval*, pages 161–176, 1994.

[5] C. Cortes and V. Vapnik. Support-vector networks. *Machine Learning*, (20):273–297, 1995.

[6] J. Cowie, E. Ludovik, and R. Zacharski. An Autonomous, Web-based, Multilingual Corpus Collection Tool. In *Proceedings of the International Conference on Natural Language Processing and Industrial Applications*, 1998.

[7] S. Dumais, J. Platt, and D. Heckerman. Inductive Learning Algorithms and Representations for Text Categorization. In *Proceedings of the 7th International Conference on Information and Knowledge Management*, 1998.

[8] T. Dunning. Statistical Identification Language. Technical Report CRL MCCS-94-273, Computing Research Lab, New Mexico State University, 1994.

[9] N. Ingle. A language identification table. *The Incorporated Linguist*, 15(4):98–101, 1991.

[10] T. Joachims. Text Categorization with Support Vector Machines: Learning with Many Relevant Features. In *Proceedings of the European Conference on Machine Learning*, 1998.

[11] S. Kulikowski. Using short words: a language identification algorithm. Technical Report, 1991.

[12] E. Osuna, R. Freund, and F. Girosi. Training Support Vector Machines: An application to face detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 130–136, 1997.

[13] J. Platt. Fast training of SVMs using sequential minimal optimization. In B. Scholkopf, C. Burges, and A. Smola, editors, *Advances in Kernel Methods-Support Vector Learning*. MIT Press, 1998.

[14] M. Porter. An algorithm for suffix stripping. *Program (Automated Library and Information Systems)*, 14(3):130–137, 1980.

[15] V. Vapnik. *Statistical Learning Theory*. John Wiley and Sons, 1998.